fintech_ devcon

PRESENTED BY MOOV

<iframe>able finance:

designing and building embedded UIs

<ifr

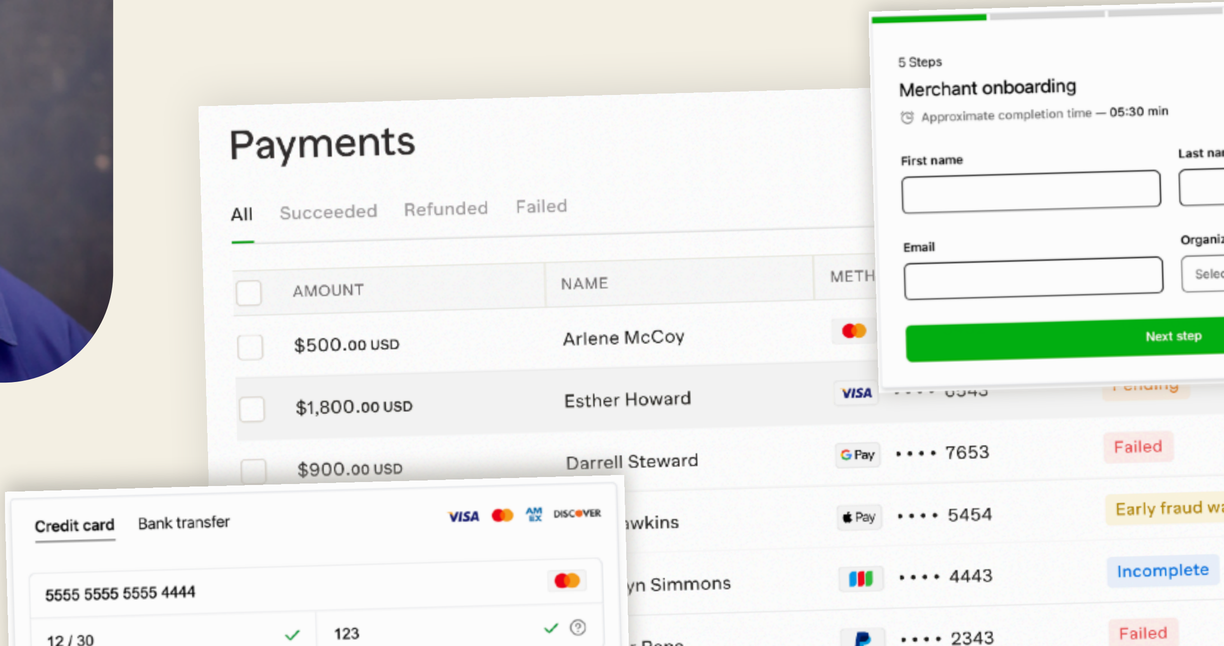
desigr

<iframe>able finance:

designing and building embedded UIs

Adam Solove

Product @ Rainforest



5 Steps

Merchant onboarding

Approximate completion time — 05:30 min

First name

Last na

Email

Organiz

Selec

Next step

Payments

All Succeeded Refunded Failed

	AMOUNT	NAME	METH		
	$500.00 USD	Arlene McCoy			
	$1,800.00 USD	Esther Howard	VISA	•••• 6543	Pending
	$900.00 USD	Darrell Steward	G Pay	•••• 7653	Failed
		awkins	Pay	•••• 5454	Early fraud wa
		yn Simmons		•••• 4443	Incomplete
		Pena		•••• 2343	Failed

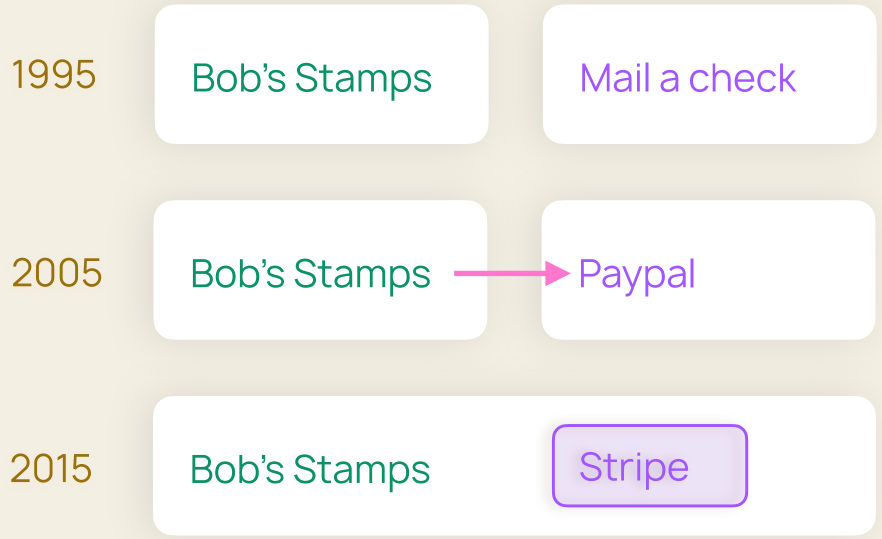
Credit card Bank transfer

VISA AM EX DISCOVER

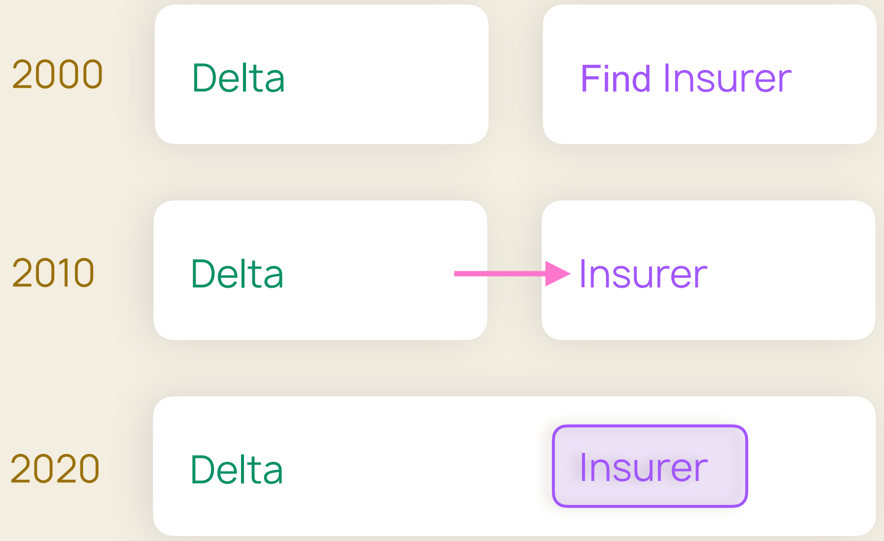
5555 5555 5555 4444

12 / 30 ✓ 123 ✓

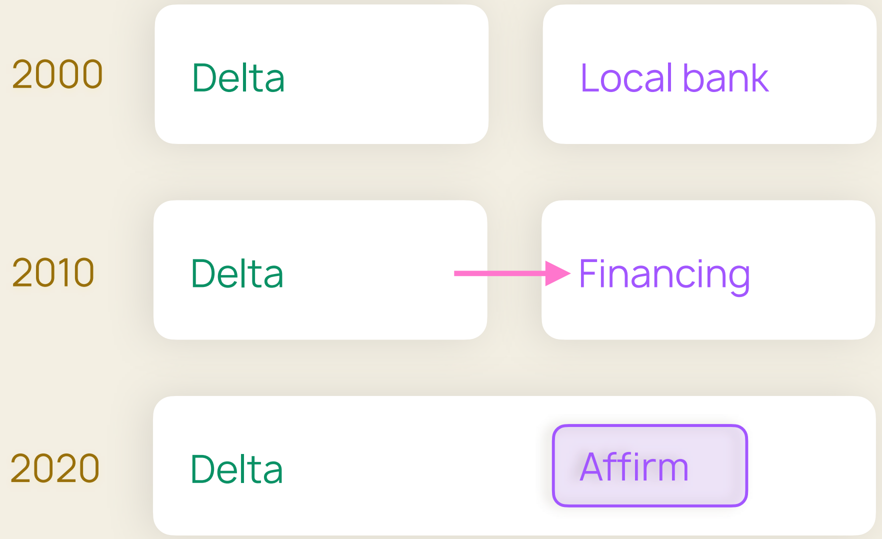
<iframe>d payments


1995
Bob's Stamps
Mail a check
2005
Bob's Stamps
Paypal
2015
Bob's Stamps
Stripe

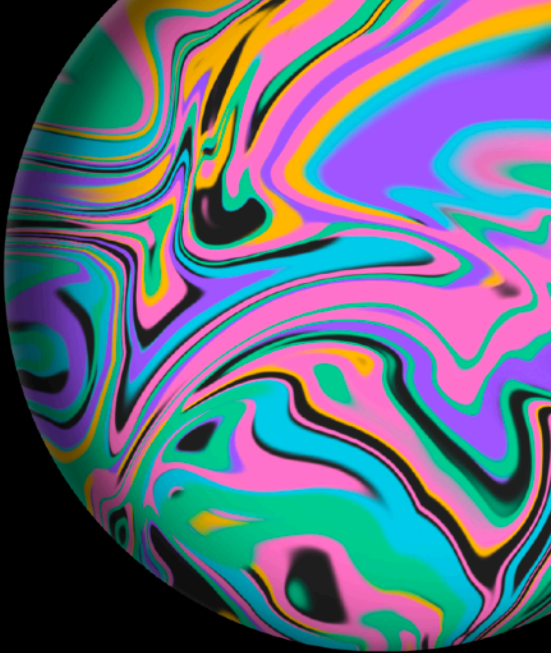
<iframe>d insurance


2000
Delta
Find Insurer
2010
Delta
Insurer
2020
Delta
Insurer

`<iframe>d` credit


2000
Delta
Local bank
2010
Delta
Financing
2020
Delta
Affirm

Great embedded UIs
make great distribution
make winners and losers



Contents

Product design

Stakeholders & constraints

Mindset

Holistic thinking

Communicating

Product mindset for embedded UIs

Get ready for more

- collaboration
- stakeholders and constraints
- obscure technical limitations
- security and compliance concerns

More stakeholders, more constraints

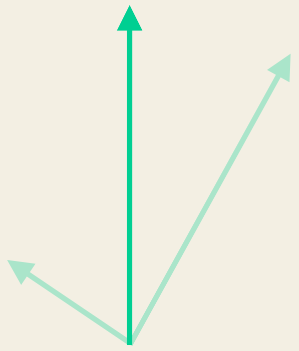
Normal constraints

- Good business
- Usable for customers
- Fast, Reliable
- Secure
- Compliant, Legal

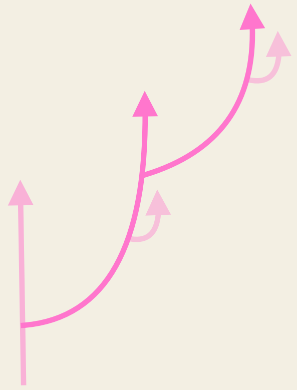
Extra constraints

- Good for partner's business
- Partner PMs' and designers' opinions
- Partner's engineers and existing systems
- Partner's security and compliance team
- Checking partner's compliance

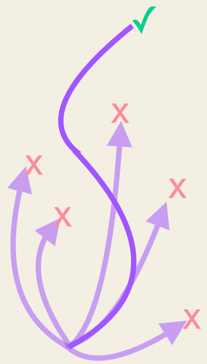
Different product mindsets



Guess a direction and move fast



Experiment by following data

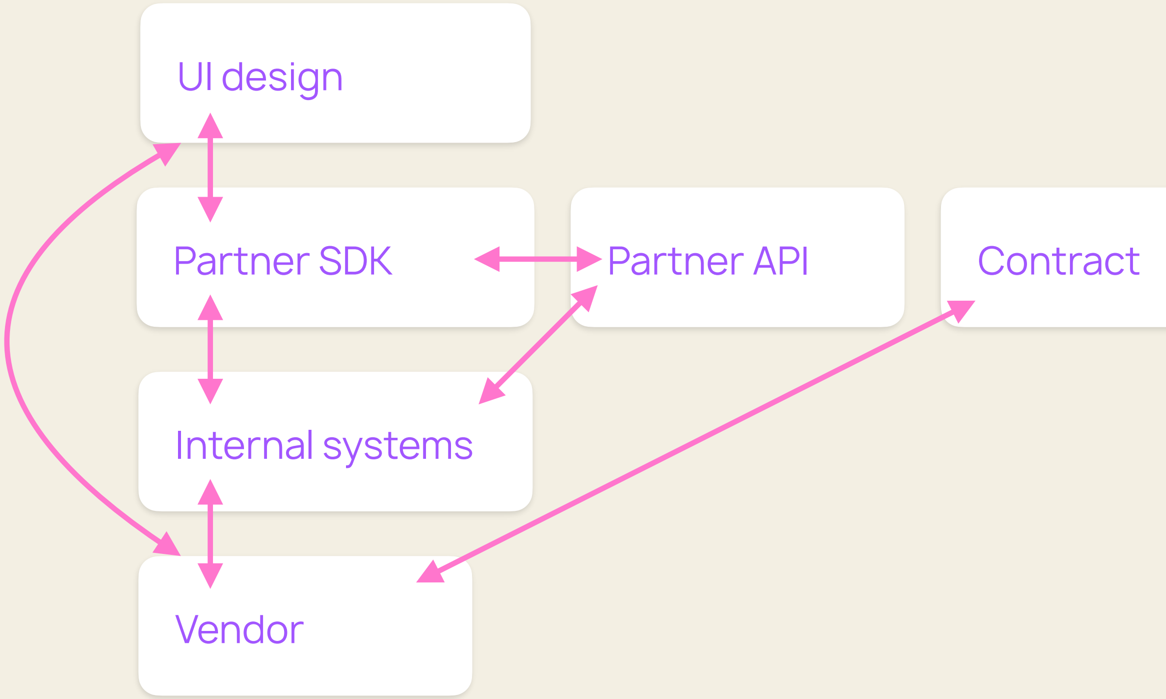


Navigate through a minefield

Think holistically

Hold the vision

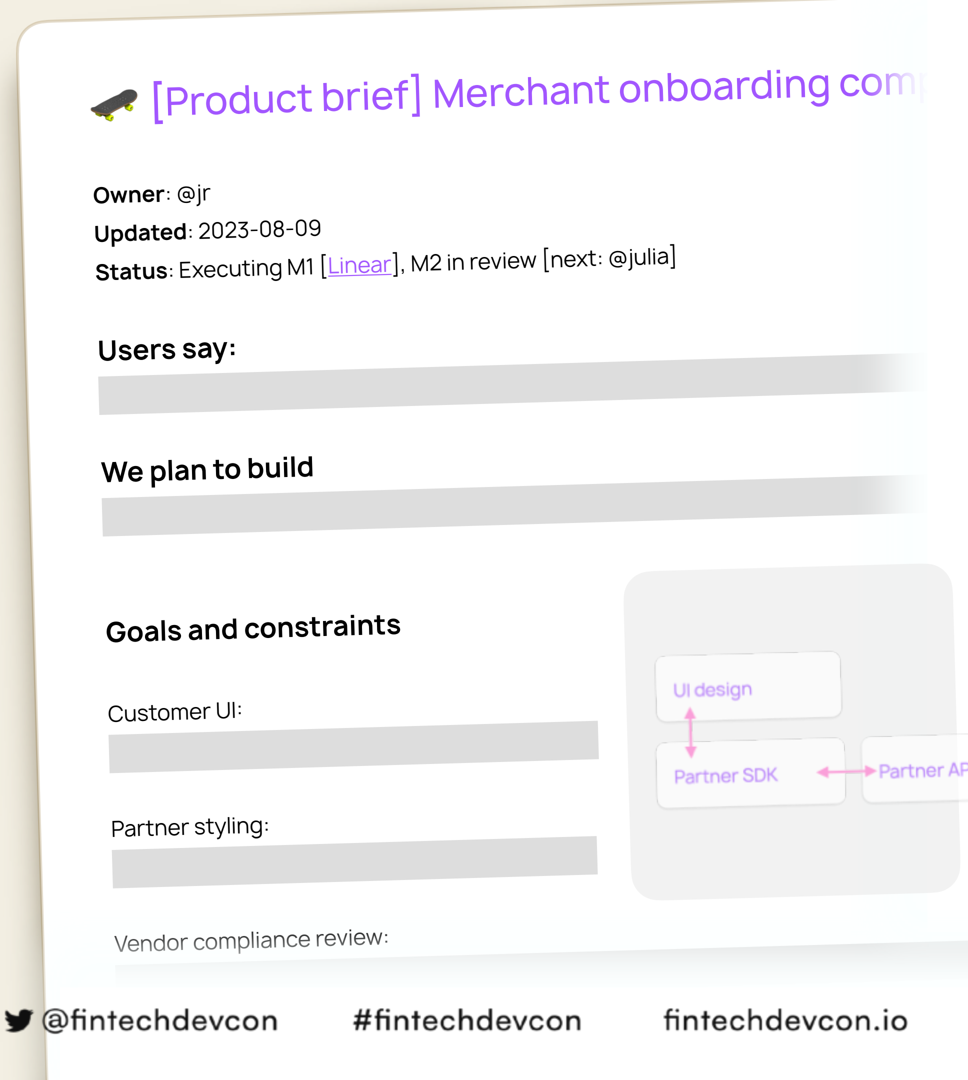
- Made tradeoffs across levels
- Track down the implications
- Explain what can't change



Write it down

Share the vision

- A working document, not a spec
- Thinking happens **in** and **with** the doc
- Zoom to different levels of detail
- Sharable across teams



[Product brief] Merchant onboarding com

Owner: @jr
Updated: 2023-08-09
Status: Executing M1 [Linear], M2 in review [next: @julia]

Users say:

We plan to build

Goals and constraints

Customer UI:

Partner styling:

Vendor compliance review:

Developer experience

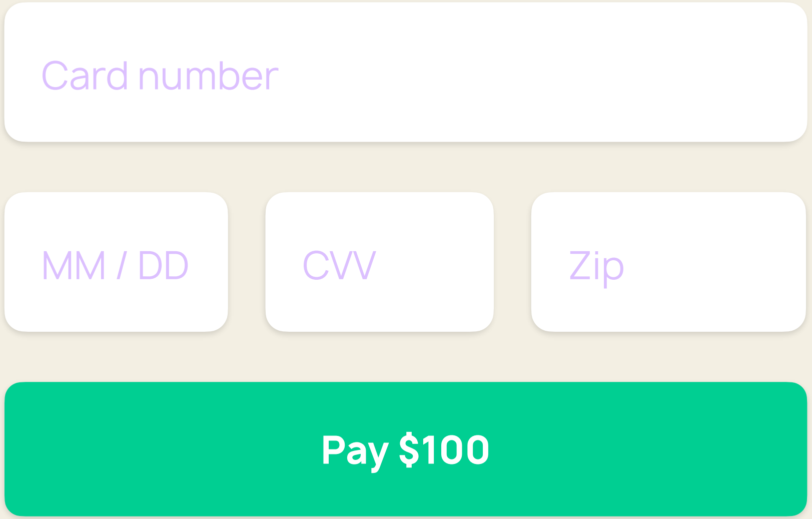
Control flow

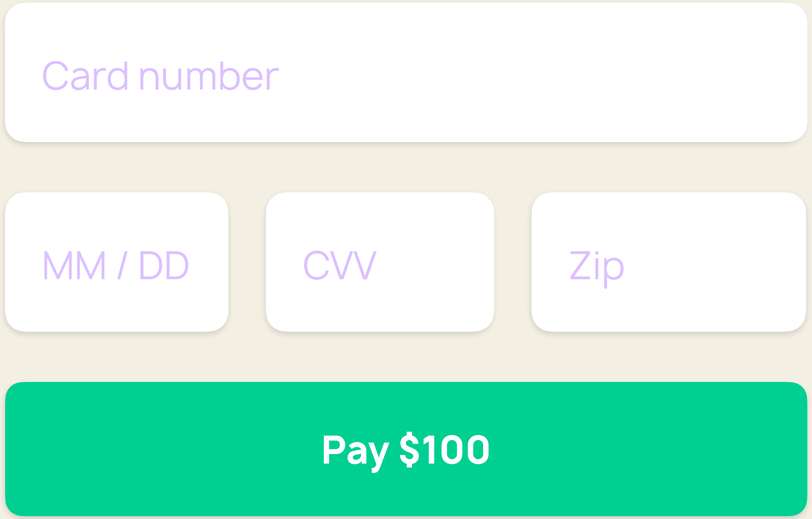
API design

Styling

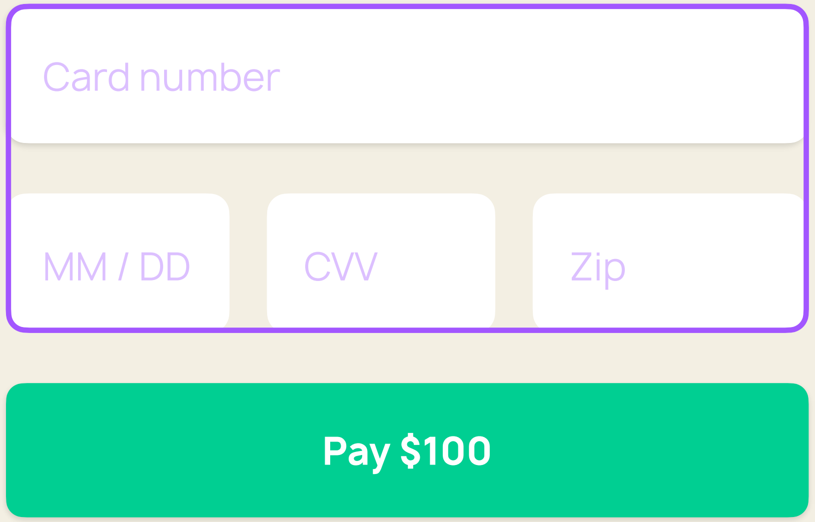
Quality review

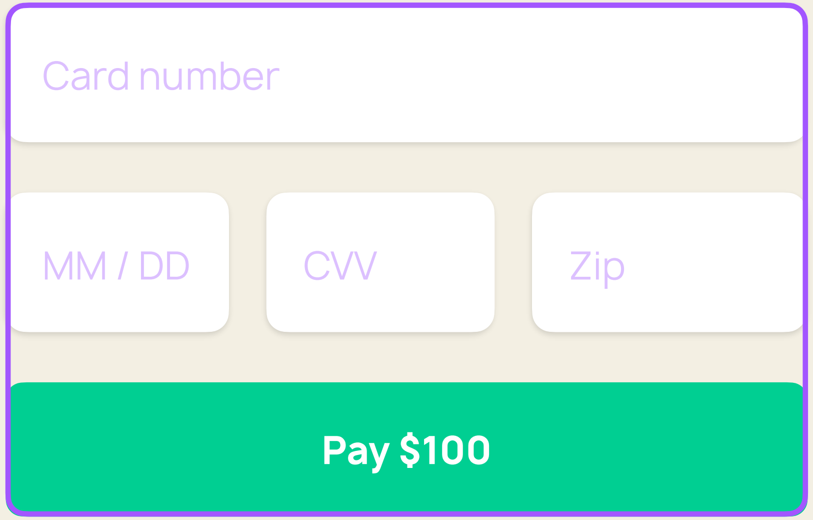
Control flow is the hard part





Control flow is the hard part



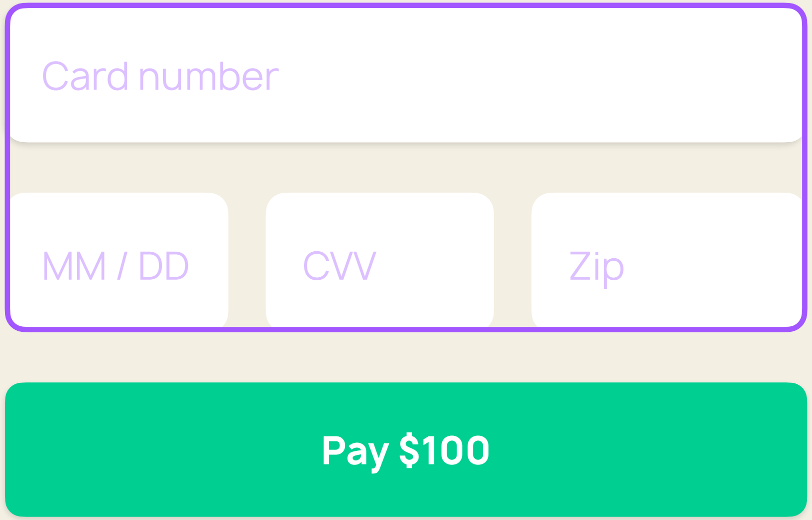


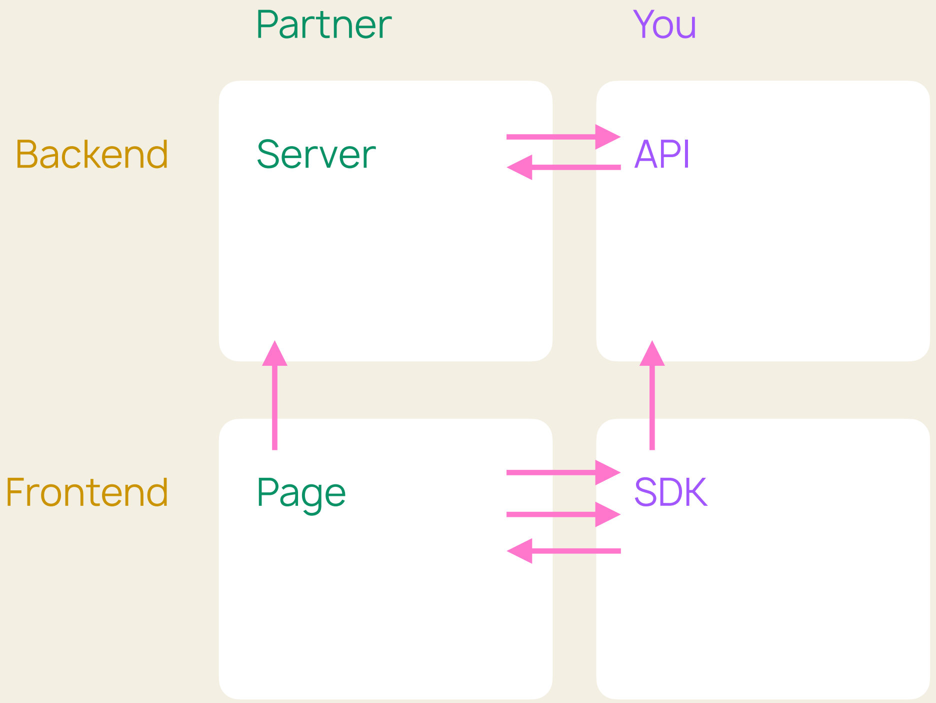
Control flow

Four-square diagram

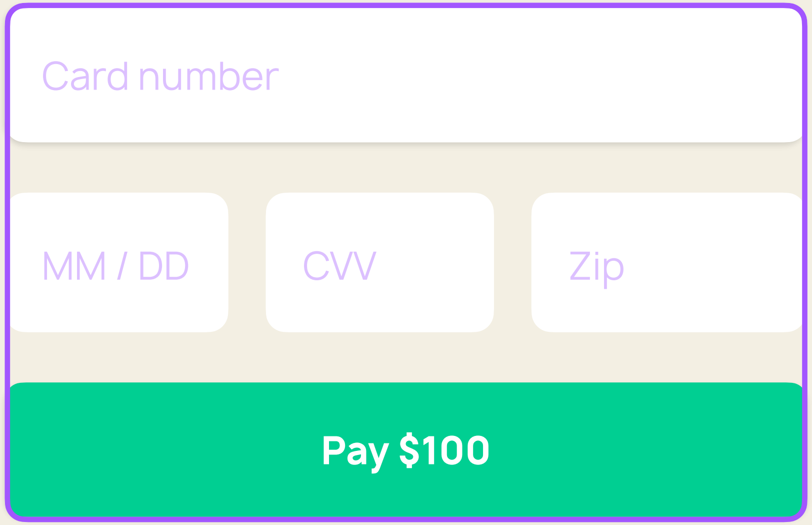
	Partner	You
Backend	Server	API
Frontend	Page	SDK

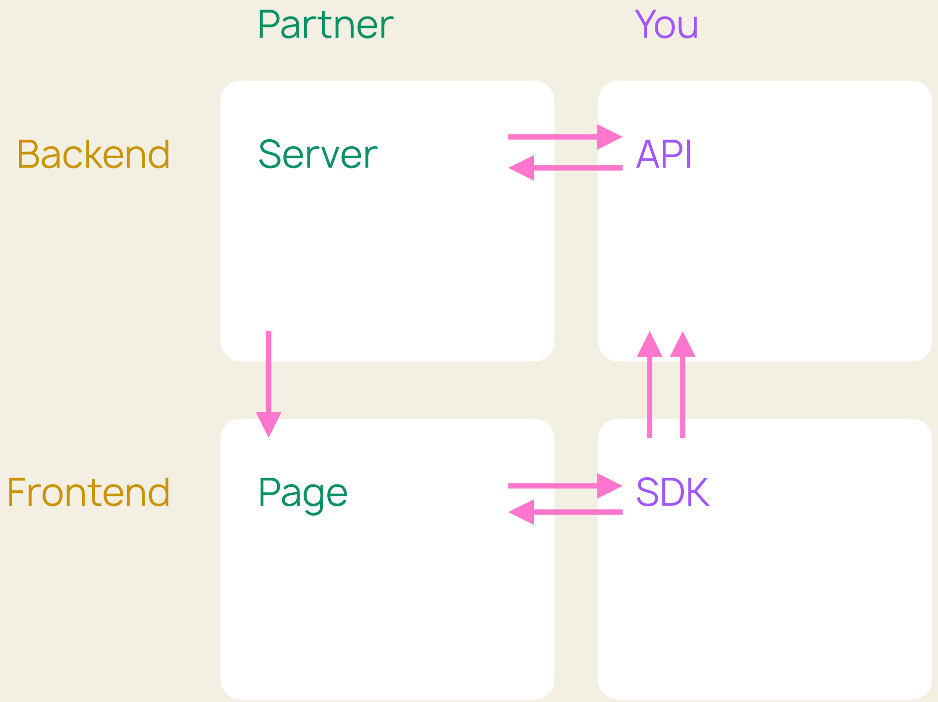
Flow one


Card number
MM / DD
CVV
Zip
Pay $100

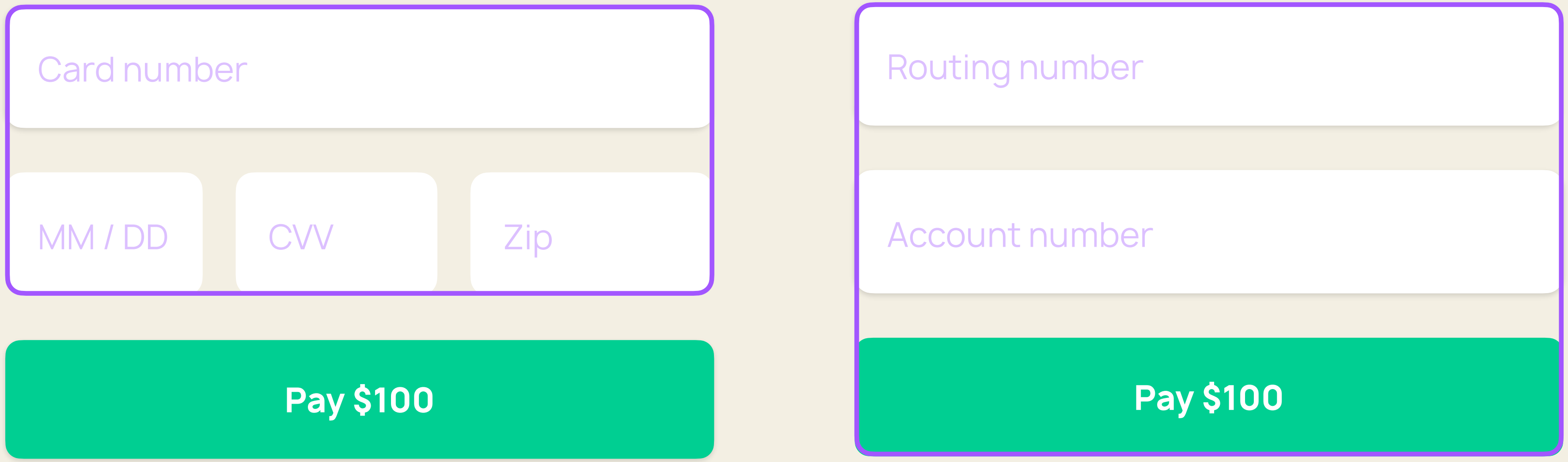

Partner
You
Backend
Server
API
Frontend
Page
SDK

Flow two


Card number
MM / DD
CVV
Zip
Pay $100


Partner
You
Backend
Frontend
Server
API
Page
SDK

Control flow tradeoffs



Card number

MM / DD CVV Zip

Pay $100

Routing number

Account number

Pay $100

Frontend APIs

Make it easy and universal

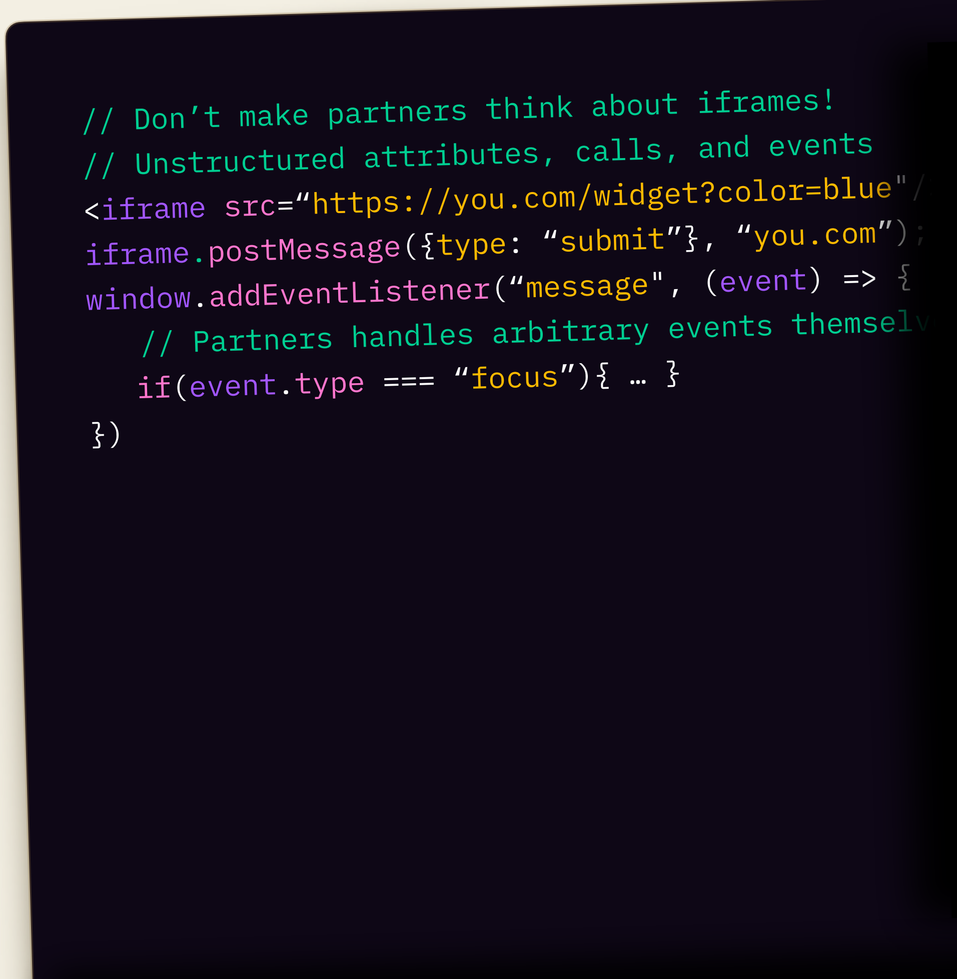
Developer experience

Frontend APIs

Make it easy and universal

Your partners:

- Hate and fear `<iframe>`s



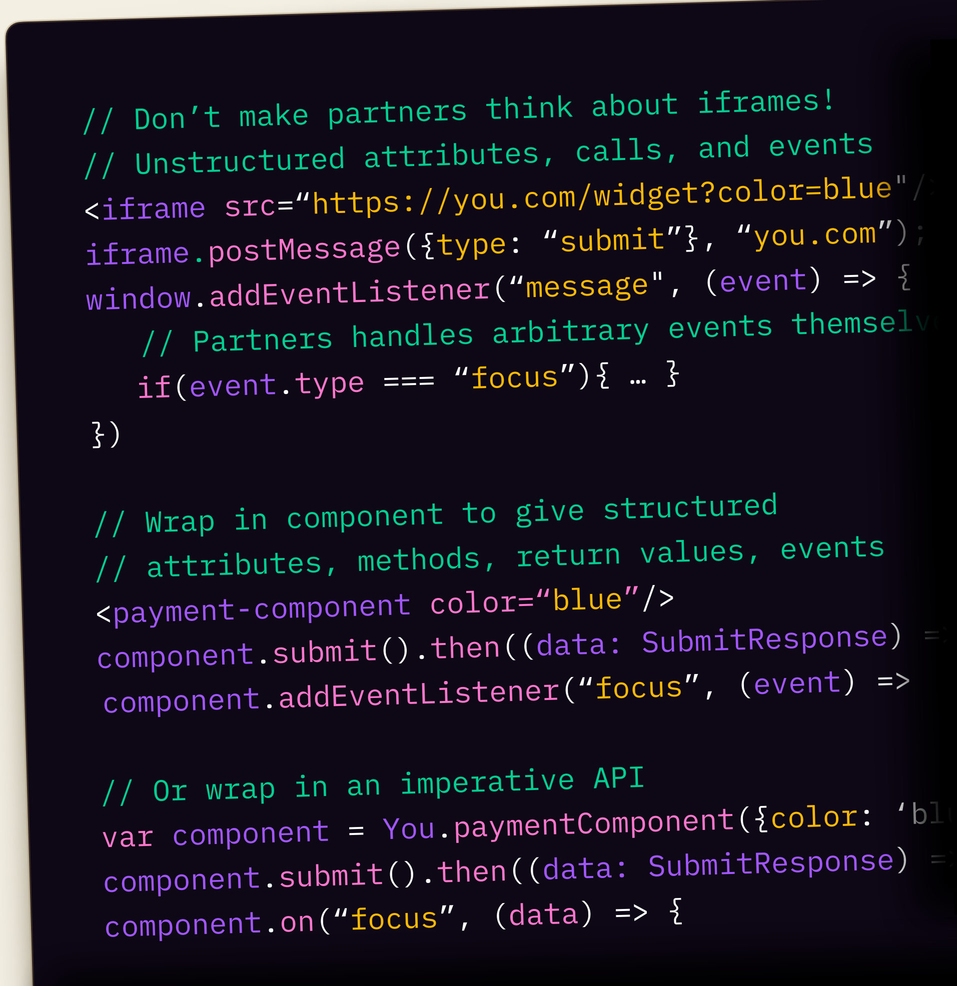
```
// Don’t make partners think about iframes!
// Unstructured attributes, calls, and events
<iframe src=“https://you.com/widget?color=blue"/
iframe.postMessage({type: “submit”}, “you.com”);
window.addEventListener(“message", (event) => {
   // Partners handles arbitrary events themselv
   if(event.type === “focus”){ … }
})
```

Frontend APIs

Make it easy and universal

Your partners' developers:

- Hate and fear `<iframe>`s
- Want types, auto-complete, lint, or any other help to use your API



```
// Don't make partners think about iframes!
// Unstructured attributes, calls, and events
<iframe src="https://you.com/widget?color=blue"/
iframe.postMessage({type: "submit"}, "you.com");
window.addEventListener("message", (event) => {
   // Partners handles arbitrary events themsel
   if(event.type === "focus"){ … }
})

// Wrap in component to give structured
// attributes, methods, return values, events
<payment-component color="blue"/>
component.submit().then((data: SubmitResponse) =
component.addEventListener("focus", (event) =>

// Or wrap in an imperative API
var component = You.paymentComponent({color: 'bl
component.submit().then((data: SubmitResponse) =
component.on("focus", (data) => {
```

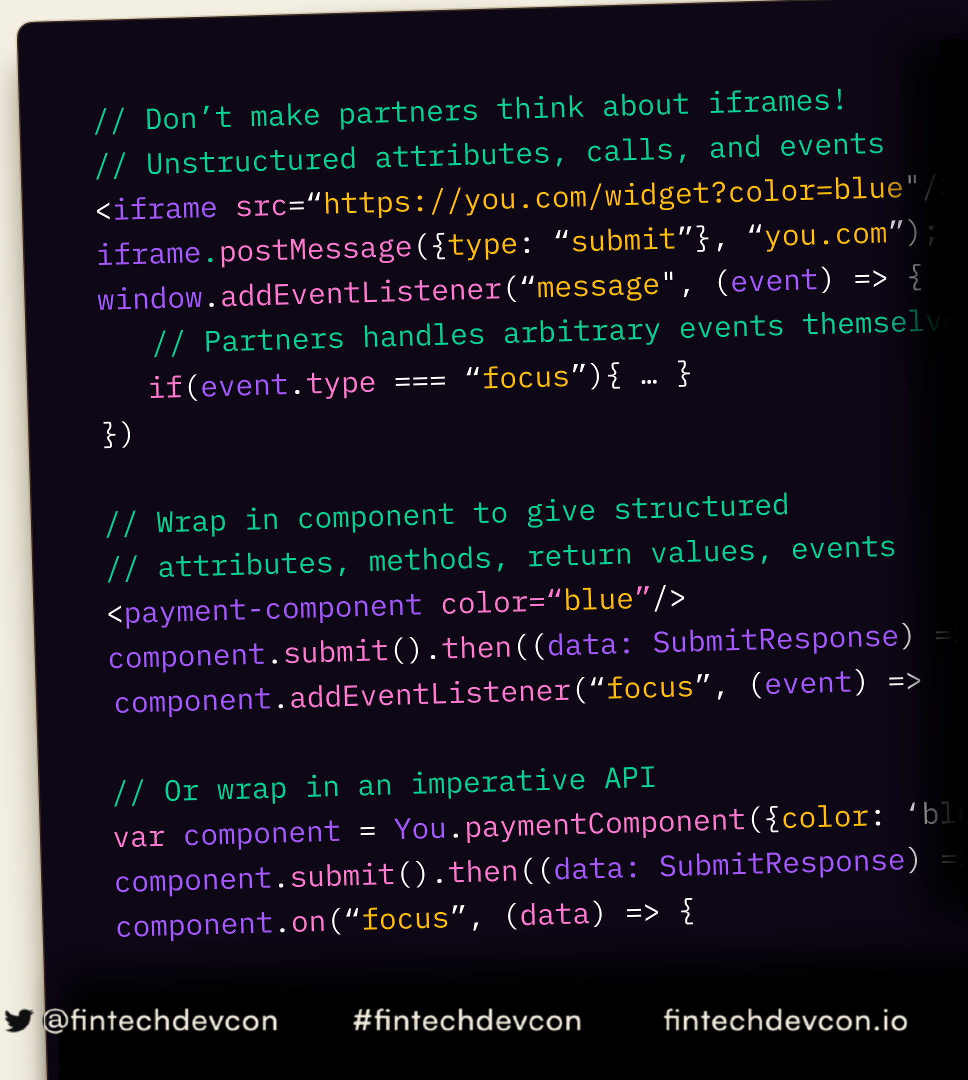
Developer experience

Frontend APIs

Make it easy and universal

Your partners' developers:

- Hate and fear `<iframe>`s
- Want types, auto-complete, lint, or any other help to use your API
- Use a different framework, language, design system, and build tool

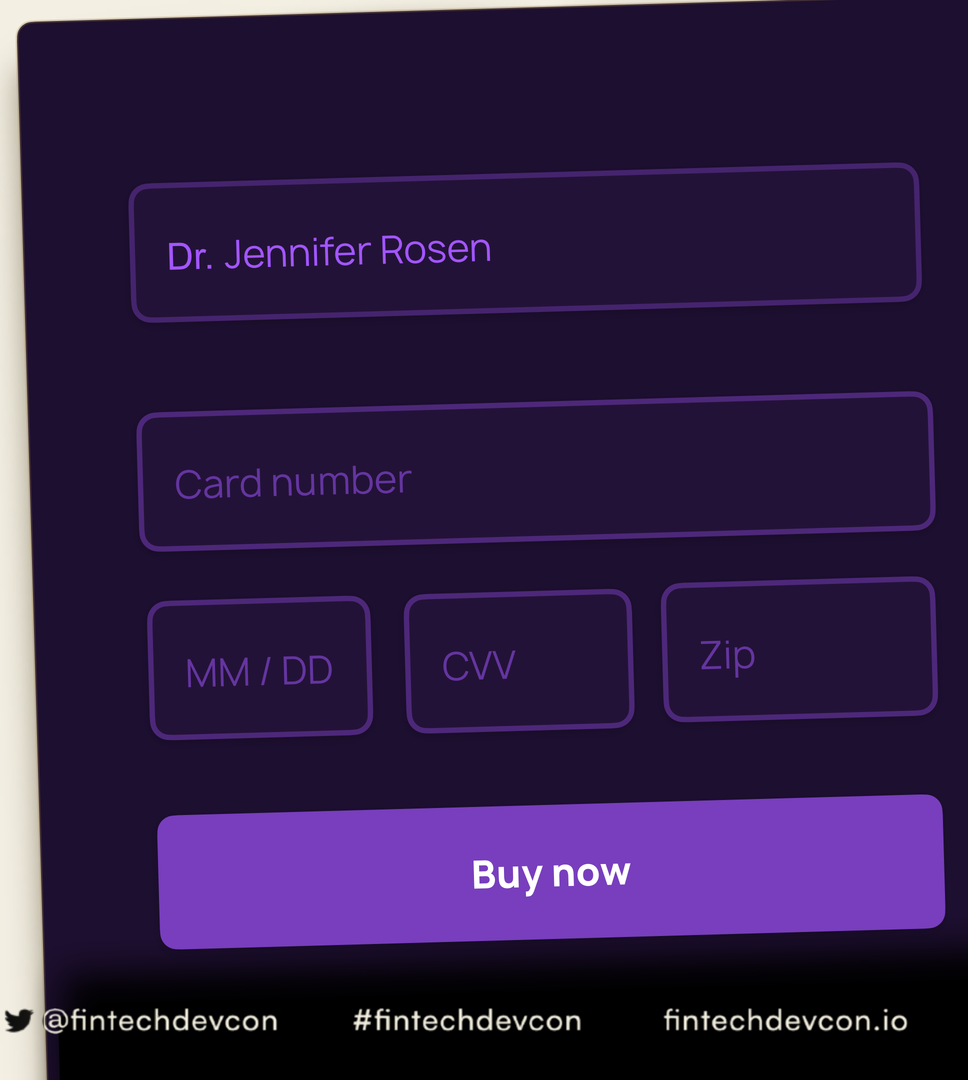


```
// Don't make partners think about iframes!
// Unstructured attributes, calls, and events
<iframe src="https://you.com/widget?color=blue"
iframe.postMessage({type: "submit"}, "you.com")
window.addEventListener("message", (event) => {
   // Partners handles arbitrary events themsel
   if(event.type === "focus"){ … }
})

// Wrap in component to give structured
// attributes, methods, return values, events
<payment-component color="blue"/>
component.submit().then((data: SubmitResponse)
component.addEventListener("focus", (event) =>

// Or wrap in an imperative API
var component = You.paymentComponent({color: 'bl
component.submit().then((data: SubmitResponse)
component.on("focus", (data) => {
```

Styling APIs

Work backwards from your goal

- Take pages from your prospective and aspirational partners
- Design how your embedded UI would best fit into their page
- Then built in the style options



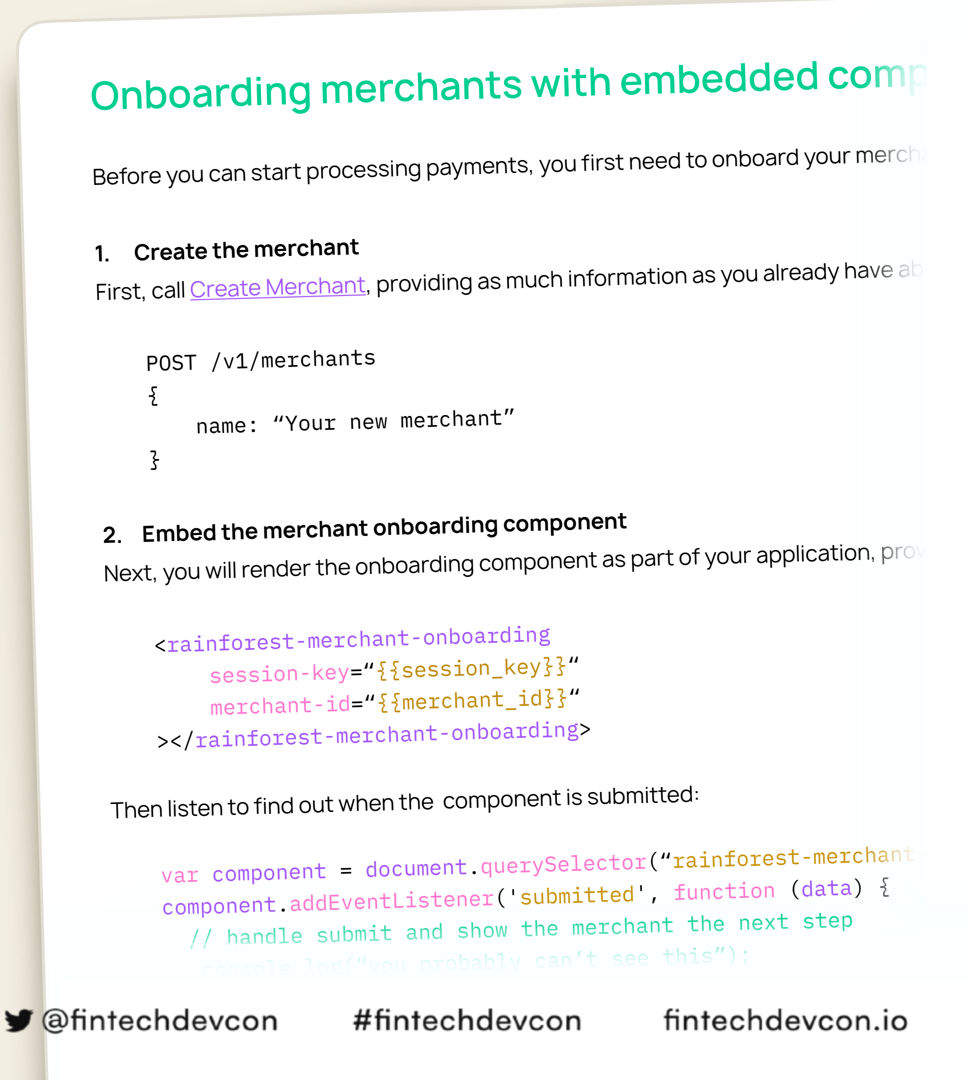
Developer experience

Draft docs

Will it make sense to a partner?

Write a draft integration guide for the simplest complete use-case:

- Explain the feature
- List the setup requirements, API calls, events, and webhooks.
- Show screenshots or sample results
- Explain how to test in development



Onboarding merchants with embedded comp

Before you can start processing payments, you first need to onboard your merch

1. **Create the merchant**

First, call Create Merchant, providing as much information as you already have ab

```
POST /v1/merchants
{
    name: "Your new merchant"
}
```

2. **Embed the merchant onboarding component**

Next, you will render the onboarding component as part of your application, pro

```
<rainforest-merchant-onboarding
    session-key="{{session_key}}"
    merchant-id="{{merchant_id}}"
></rainforest-merchant-onboarding>
```

Then listen to find out when the component is submitted:

```
var component = document.querySelector("rainforest-merchant
component.addEventListener('submitted', function (data) {
  // handle submit and show the merchant the next step
  console.log("you probably can't see this");
```

Friction logs

Experience your product as an outsider

- **What?** Try it as if an external user
- **Who?** You, your company, real users
- **When?** Before, during, and after

Do something unusual: use a different language, programming environment, browser, business model, etc.

2022

Criticism and discomfort

- You're delivering tough news on someone's work
- Emotional intelligence is a delicate balance
- Giving feedback is hard, receiving is harder

fintech_ devcon

fintechdevcon.io @fintechdevcon

The developer feedback experience: Friction logging

Mike Bifulco

Stripe

Engineering

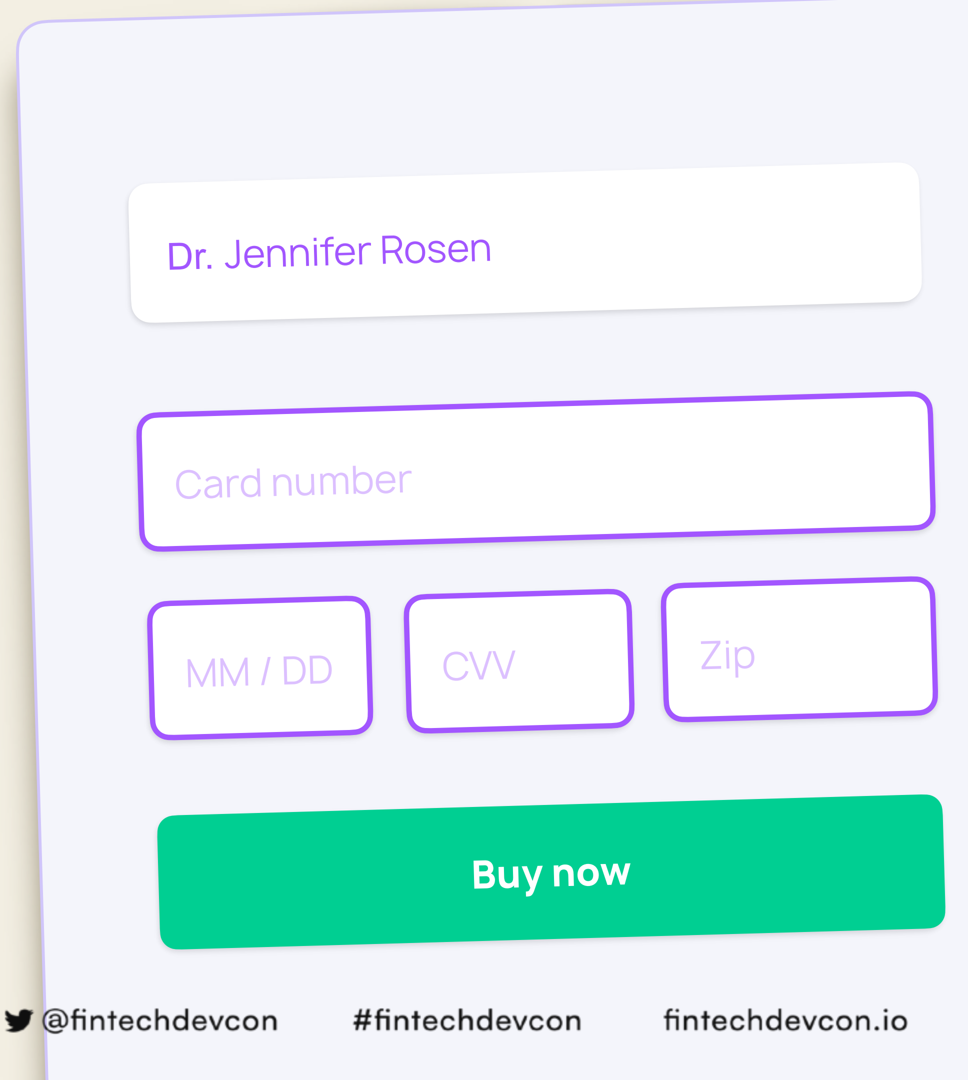
`<iframe>` **limits**

`<iframe>` **communication**

Security

`<iframe>` limits

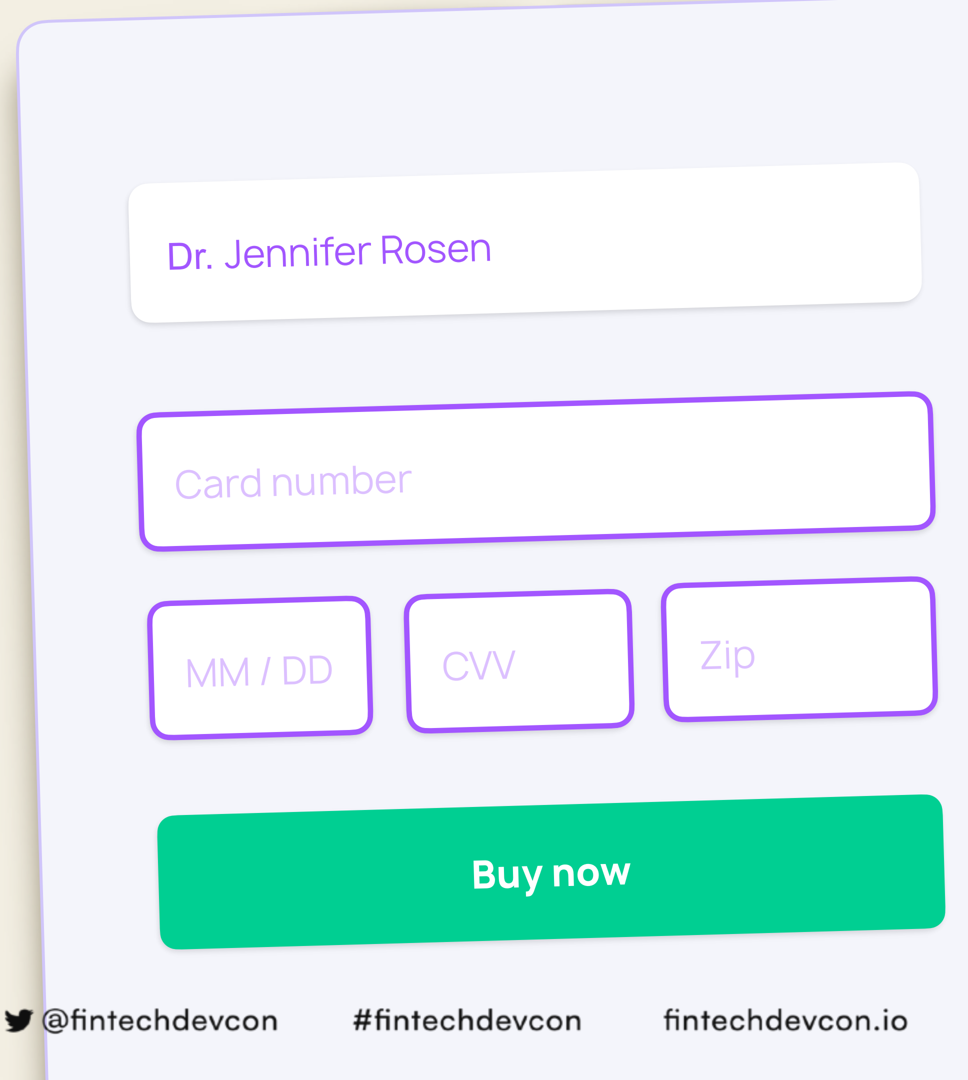
Don't assume it will "just work"



`<iframe>` limits

Don't assume it will "just work"

- Focus and blur
- Autocomplete
- User activation



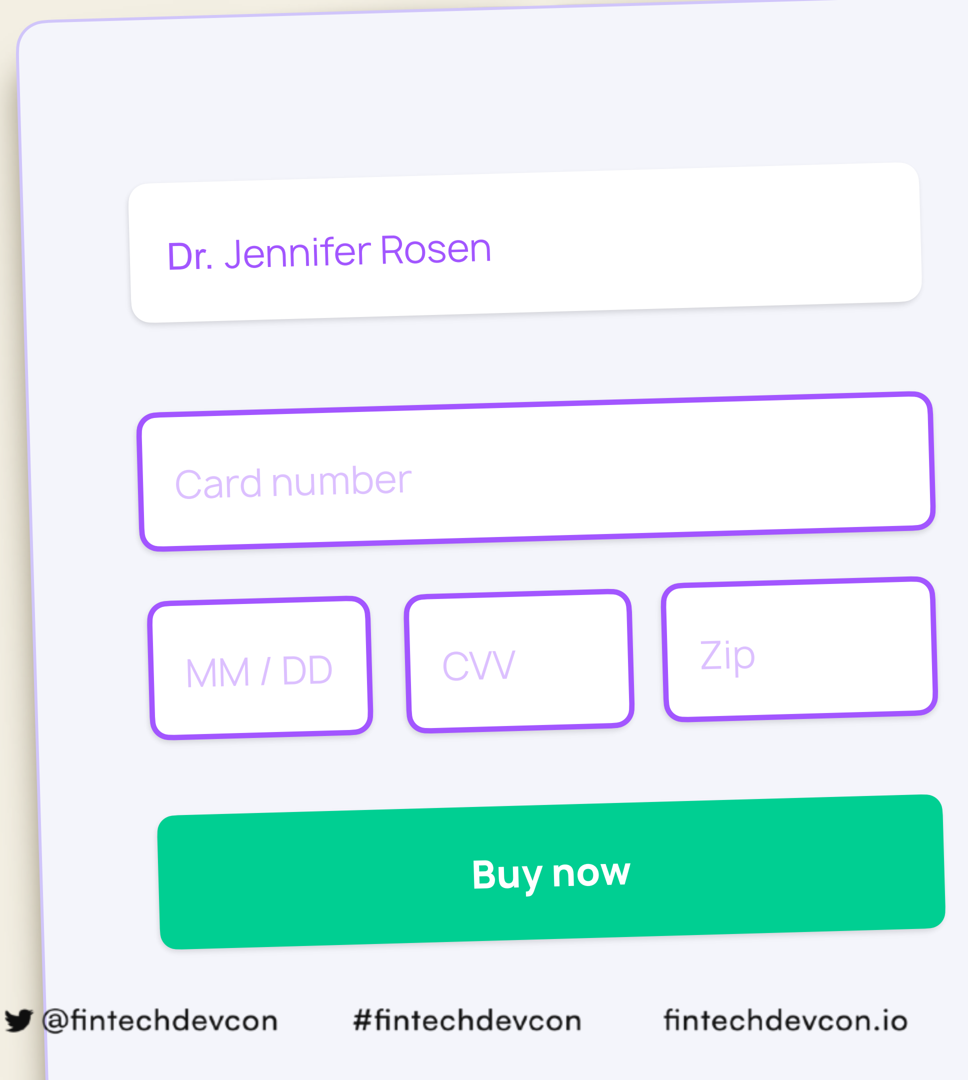
`<iframe>` limits

Don’t assume it will “just work”

- Focus and blur
- Autocomplete
- User activation

Develop and test realistically

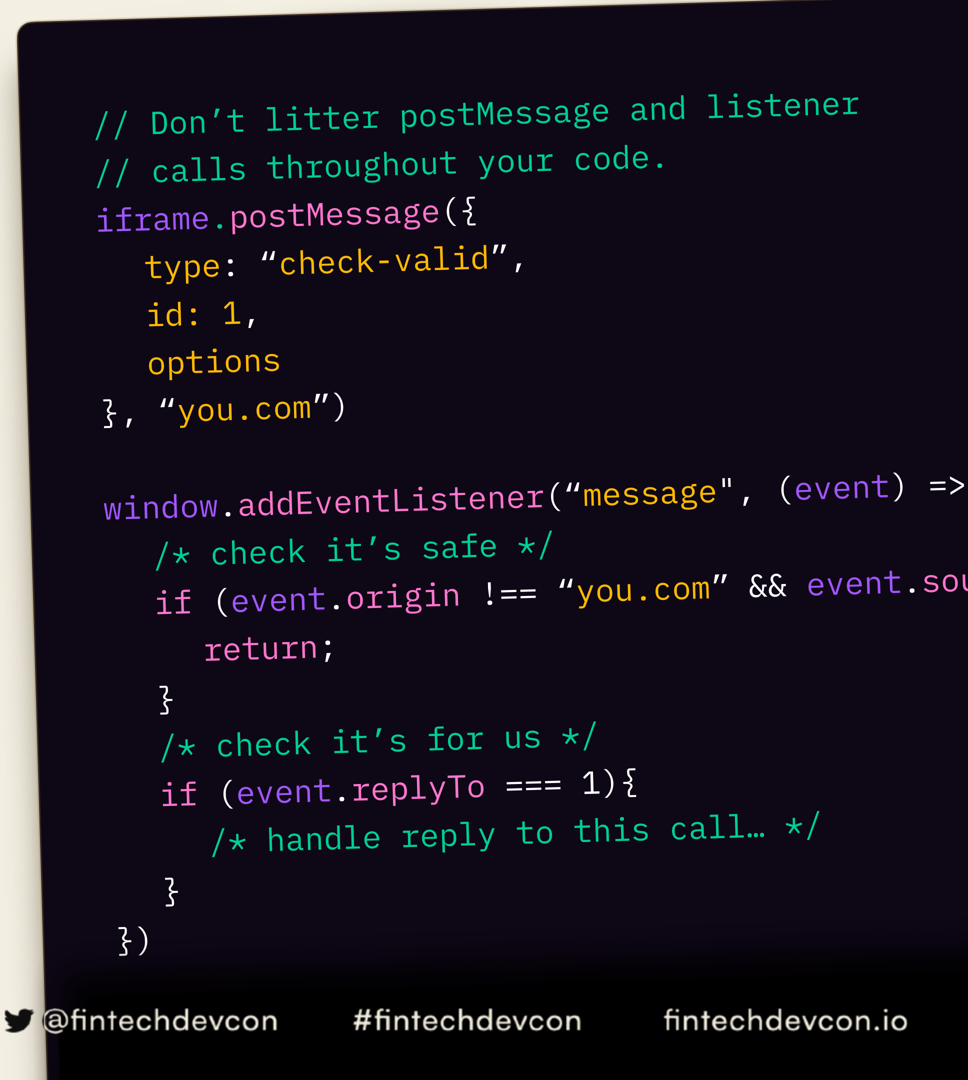
- In an iframe, on https, from another origin
- Using a realistic sample integration



Engineering

<iframe> communication

Wrap postMessage() to make it better



```
// Don’t litter postMessage and listener
// calls throughout your code.
iframe.postMessage({
   type: “check-valid”,
   id: 1,
   options
}, “you.com”)

window.addEventListener(“message", (event) =>
   /* check it’s safe */
   if (event.origin !== “you.com” && event.sou
      return;
   }
   /* check it’s for us */
   if (event.replyTo === 1){
      /* handle reply to this call… */
   }
})
```

`<iframe>` communication

Wrap `postMessage()` to make it better

- Add type or validity checking to both sides
- Add safety checks in one place
- Create a request-response `Promise` flow
- Debug-level logging for payloads

```
// Build a nicer abstraction
child
  .checkValid(options as CheckValidRequest)
  .then((data: CheckValidResponse) => {
    /* handle response */
  })
```

<iframe> security

<iframe>s aren't enough

`<iframe>` security

`<iframe>`s aren't enough

- Check message provenance

<iframe> security

`<iframe>`s aren’t enough

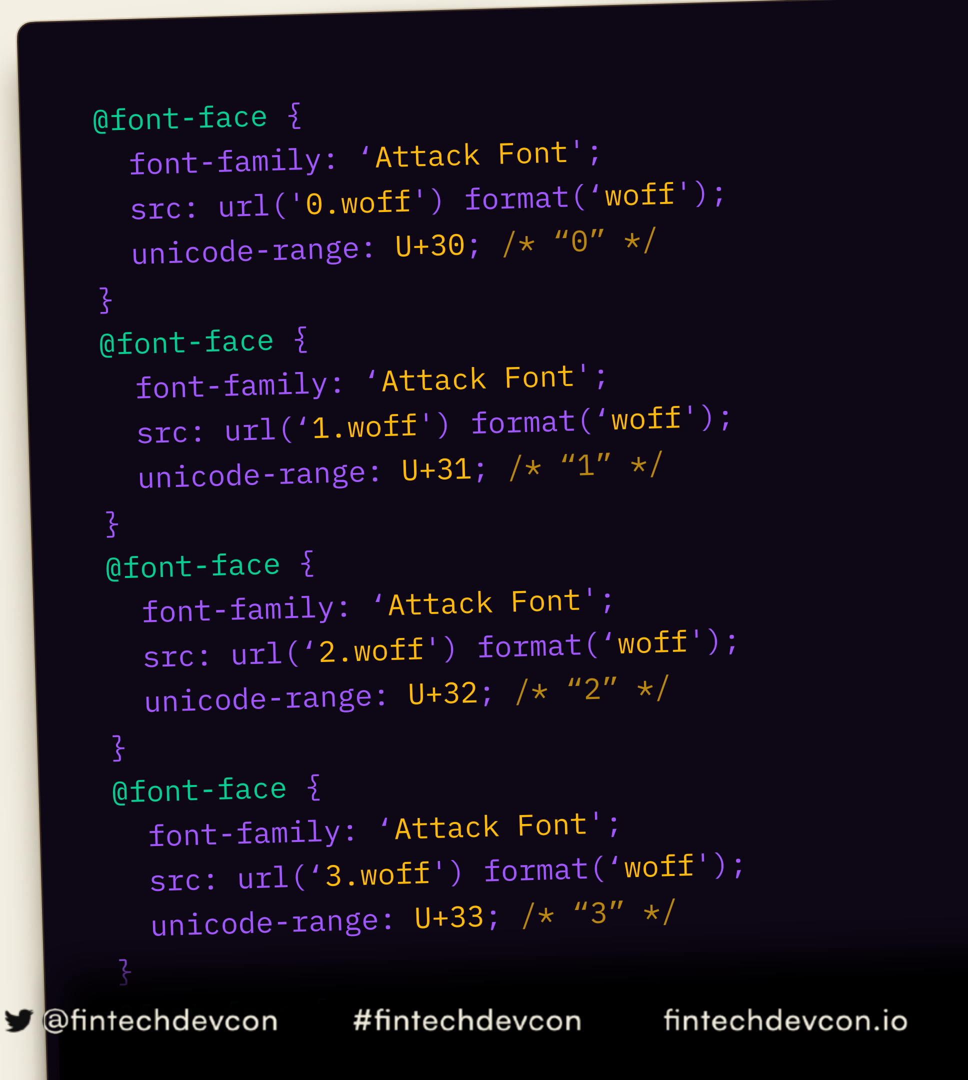
- Check message provenance
- Use Content Security Policy

```
Content-Security-Policy:
   default-src ‘self’;
   img-src static.you.com;
   style-src static.you.com;
   script-src js.you.com;
   connect-src api.you.com;
   report-to https://ops.you.com/collector
```

<iframe> security

<iframe>s aren't enough

- Check message provenance
- Use Content Security Policy
- Watch for hidden side-effects



```
@font-face {
  font-family: 'Attack Font';
  src: url('0.woff') format('woff');
  unicode-range: U+30; /* "0" */
}
@font-face {
  font-family: 'Attack Font';
  src: url('1.woff') format('woff');
  unicode-range: U+31; /* "1" */
}
@font-face {
  font-family: 'Attack Font';
  src: url('2.woff') format('woff');
  unicode-range: U+32; /* "2" */
}
@font-face {
  font-family: 'Attack Font';
  src: url('3.woff') format('woff');
  unicode-range: U+33; /* "3" */
}
```

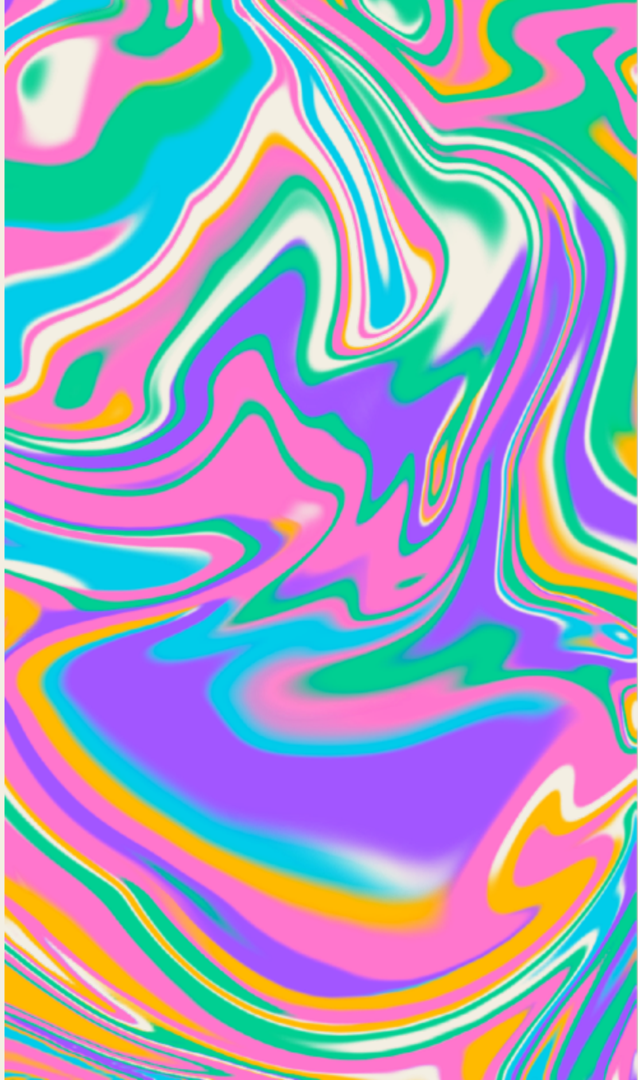
Raise our ambitions

Embedded UIs can be easy, beautiful, and profitable.

`<iframe>` friends, let’s talk!

Find me at the conference. Or talk later:

- adam.solove@rainforestpay.com
- linkedin.com/in/asolove
- adamsolove.com



Thank *you.*

Adam Solove, Product @ Rainforest

<iframe>able finance:

designing and building embeddable UIs